The FreeBSD Compact Guide

Third Edition

Written By

David Childers

www.ScenicRadio.Com

Relaxing Entertainment For The World

www.BroadcastingWorld.Com

Global Broadcast Information Portal

The FreeBSD Documentation License

Creative Common License

Cover graphic - This picture was taken from the Dictionnaire encyclopédique Trousset, also known as the Trousset encyclopedia, Paris, 1886 - 1891.
www.oldbookillustrations.com/pages/vignette1.php?lng=en

Foreword graphic - This picture was taken from “Safe Counsel: Searchlights on Health: Light on Dark Corners” (1896)
www.fromoldbooks.org/pictures-of-old-books/pages/cimg5129-rose-on-music-book-on-piano/800x600-q85.html#details

About The Author

David Childers

David Childers is the Content Manager for the Global Broadcasting portal www.BroadcastingWorld.com. He is very active in the Internet broadcast industry and has published numerous guides and a book about this growing technological field. He is also the webmaster of www.ScenicRadio.com, the global destination for relaxing entertainment. He is also a member of the International Association of Internet Broadcasters and the American Communications Association.

Mr. Childers' work has been cited in several national and International publications, including these:

Five Essays on Copyright In the Digital Era
Turre Publishing

Research On High-Profile Digital Video Production
Digital Content Association of Japan

Video Podcasting in Perspective: The History, Technology, Aesthetics and Instructional Uses of a New Medium
Journal of Educational Technology Systems

Video Podcasting: When, Where and How it's Currently used for Instruction
The National Convention of the Association for Educational Communications and Technology

Preservation of audiovisual mediums: Problems and challenges
Platform for Archiving and Preservation of Art on Electronic and Digital Media
Centre of Expertise in Digital Heritage

P2P Technology Trend and Application to Home Network
Electronics and Telecommunications Research Institute Journal

Peer To Peer Computing - The Evolution of a Disruptive Technology
Idea Group Publishing

Peer-to-Peer Systems and Applications
Lecture Notes In Computer Science
Springer Berlin / Heidelberg

Foreword

I have updated and expanded this guide so that it continues to remain a valuable resource for the FreeBSD community.

I would like to thank Scarlet Coker for providing assistance with the editing of the manuscript and James Davey at Broadcasting World for allowing me the opportunity to create this guide.

It is my sincere hope that the reader finds this guide beneficial.

David Childers
www.scenicradio.com

March 2015

Posveèeno Neži Vidmar.

Pri veri se ne gre za vprašanje ali Bog obstaja ali ne.
Gre se za to, da verjamemo, da je ljubezen brez plaèila prav tako dragocena.

If you have any trouble sounding condescending, find a Unix user to show you how it's done.

Scott Adams

Feedback

Please feel free to contact the author if you have any questions or comments. Your feedback is greatly appreciated.

You can contact the author here: www.KL7AF.com

Index

What Is FreeBSD

There are a number of Unix-like operating systems based on or descended from the Berkeley Software Distribution (BSD) series of Unix variants. The three most notable descendants in current use are FreeBSD, OpenBSD, and NetBSD, which are all derived from 386BSD and 4.4BSD-Lite, by various routes.

FreeBSD

This version of BSD aims to make an operating system usable for any purpose.[4] It is intended to run a wide variety of applications, be easy to use, contain cutting edge features, and be highly scalable on very high load network servers

OpenBSD

This version of BSD aims at security, correctness, and being as free as possible.

NetBSD

This version of BSD aims to provide a freely redistributable operating system that professionals, hobbyists, and researchers can use in any manner they wish. The main focus is portability. **(1)**

FreeBSD is a free Unix-like operating system descended from Research Unix via the Berkeley Software Distribution (BSD). Although for legal reasons FreeBSD cannot use the Unix trademark, it is a direct descendant of BSD, which was historically also called "BSD Unix" or "Berkeley Unix." The first version of FreeBSD was released in 1993, and today FreeBSD is the most widely used open-source BSD distribution, accounting for more than three-quarters of all installed systems running open-source BSD derivatives.

FreeBSD versus Linux

* FreeBSD

FreeBSD offers a complete operating system. The composite software provides a kernel, device drivers and userland utilities. **(2)**

* Linux

Linux is a composite set of software instructions that form the Linux kernel. (3)

Separate software modules are required to interface with the kernel to provide the system's higher-level functionality. **(4)**

Linux is a collection of modules that work in conjunction with the kernel to form a computer operating system.

FreeBSD can be run as a server or in a desktop environment and can be installed on various types of devices.

FreeBSD Software

There are several divisions within the FreeBSD software development community. They include releases and development branches.

Releases

FreeBSD releases are classified into "Production Releases" and "Legacy Releases". Production releases are best suited to users looking for the latest new features, and legacy releases are for users wishing to stay with a more conservative upgrade strategy.

Releases are further classified by the length of time they will be supported by the Security Officer into "Normal" and "Extended" releases. **(5)**

Development Branches

FreeBSD has two development branches: FreeBSD-CURRENT and FreeBSD-STABLE.

Using FreeBSD-CURRENT

FreeBSD-CURRENT is the “bleeding edge” of FreeBSD development, and FreeBSD-CURRENT users are expected to have a high degree of technical skill. Less technical users who wish to track a development branch should track FreeBSD-STABLE instead.

FreeBSD-CURRENT is the very latest source code for FreeBSD and includes works in progress, experimental changes, and transitional mechanisms that might or might not be present in the next official release. While many FreeBSD developers compile the FreeBSD-CURRENT source code daily, there are short periods of time when the source may not be buildable. These problems are resolved as quickly as possible, but whether or not FreeBSD-CURRENT brings disaster, or new functionality can be a matter of when the source code was synced.

Using FreeBSD-STABLE

FreeBSD-STABLE is the development branch from which major releases are made. Changes go into this branch at a slower pace and with the general assumption that they have first been tested in FreeBSD-CURRENT. This is still a development branch and, at any given time, the sources for FreeBSD-STABLE may or may not be suitable for general use. It is simply another engineering development track, not a resource for end-users. Users who do not have the resources to perform testing should instead run the most recent release of FreeBSD.

Those interested in tracking or contributing to the FreeBSD development process, especially as it relates to the next release of FreeBSD, should consider following FreeBSD-STABLE. **(6)**

The FreeBSD Operating System

The FreeBSD operating system is made up of three parts; the kernel, the shell and the programs.

The Kernel

The kernel is the heart of the operating system. It allocates time and memory to programs and handles the file structure and communication between the different parts of the computer system such as the keyboard and the screen.

Outside of the Kernel, all other programs on the system are part of the userland, which means outside the kernel. The kernel shares the system between all running userland programs. The network connection, or another part of the hardware, a userland program must contact the kernel using system calls to use the keyboard. The kernel allows multiple programs to share the hardware safely. It also switches programs in and out of the processors; thus it is a "multitasking" kernel. **(7)**

The Shell

The shell is an interface between the user and the kernel. It resembles the 'dos box' that Windows displays if you run the command cmd. When a user logs in, FreeBSD checks the username and password. FreeBSD then starts another program called the shell. The shell interprets the commands the user types and transmits them to the kernel to be executed. These commands are programs.

There are numerous shell interfaces available, and they can customized by the user, and the users can use different shells on the same machine.

The shell and kernel work together like this:

* A user types cat some file to display a file.

* The shell finds the program cat.

* The shell instructs the kernel to run the program cat on some file.

* When the program finishes the kernel passes control back to the shell and displays the Unix prompt. **(8)**

Files And Processes

Everything in Unix is a file or a process. In Unix, a file is just a destination for or a source of a stream of data. Thus a printer, for example, is a file and so is the screen.

A process is a program that is currently running. So a process may be associated with a file. The file stores the instructions that are executed for that process to run.

Another way to look at it is that file is a collection of data that can be referred to by name. Files are created by users either directly (using text editors, running compilers etc.) or indirectly (by running some program - like processing a text input file to produce a formatted file for printing).

Examples of files include:

* A text document.

* A program that was written in a programming language such as C++ or Java.

* A graphic image.

* A directory is a file that contain links to other files. **(9)**

Kernel

FreeBSD has a monolithic kernel that provides a system architecture where the entire operating system is working in kernel space. Monolithic kernels segregate virtual memory, which is a memory management technique that is implemented using both hardware and software, into kernel space and user space.

* Kernel space is strictly reserved for running a privileged operating system kernel, kernel extensions, and most device drivers.

* User space is the memory area where application software and some drivers execute.

This separation primarily serves to protect data and functionality from faults (by improving fault tolerance) and malicious behaviour (by providing computer security).

The operating system runs in supervisor mode, and the applications run in user mode in a monolithic kernel.

Supervisor mode is a hardware-mediated flag that can be changed by code running in system-level software. System-level tasks or threads will have this flag set while they are running.

This flag determines whether it would be possible to execute machine code operations such as modifying registers for various descriptor tables, or performing operations such as disabling interrupts. The idea of having two different modes to operate in comes from the concept of "with more control comes more responsibility". A program in supervisor mode is trusted never to fail since a failure may cause the whole computer system to crash.

Userspace applications do not have a hardware-mediated flag that can be changed by code running in system-level software.

The monolithic model differs from other operating system architectures in that it alone defines a high-level virtual interface over computer hardware. A set of primitives or system calls implement all operating system services such as process management, concurrency, and memory management. Device drivers can be added to the kernel as modules. **(10)**

Directory Structure

Directory	Description
/	The root directory of the file system.
/bin/	User utilities which are fundamental to both single-user and multi-user environments.
/boot/	Programs and configuration files used during operating system bootstrap.
/boot/defaults/	Default bootstrapping configuration files.
/dev/	Device nodes.
/etc/	System configuration files and scripts.
/etc/defaults/	Default system configuration files.
/etc/mail/	Configuration files for mail transport agents.
/etc/namedb/	Named configuration files.
/etc/periodic/	Scripts that are run daily, weekly, and monthly, via cron.
/etc/ppp/	ppp configuration files.
/mnt/	Empty directory commonly used by system administrators as a temporary mount point.
/proc/	Process file system.
/rescue/	Statically linked programs for emergency recovery.
/root/	The home directory for the root account.
/sbin/	System programs and administration utilities fundamental to both single-user and multi-user environments.
/tmp/	Contains temporary files. The contents of /tmp are usually NOT preserved across a system reboot. A memory-based file system is often mounted at /tmp.
/usr/	Location of the majority of user utilities and applications.
/usr/bin/	Common utilities, programming tools, and applications.
/usr/include/	Standard C include files.
/usr/lib/	Archive libraries.
/usr/libdata/	Miscellaneous utility data files.
/usr/libexec/	System daemons & system utilities (executed by other programs).
/usr/local/	Local executables, libraries, etc. Also used as the default destination for the FreeBSD ports framework. Within /usr/local, the general layout sketched out by hier for /usr should be used. Exceptions are the man directory, which is directly under /usr/local rather than under /usr/local/share, and the ports documentation is in share/doc/port.
/usr/obj/	Architecture-specific target tree produced by building the /usr/src tree.
/usr/ports/	The FreeBSD Ports Collection (optional).
/usr/sbin/	System daemons & system utilities (executed by users).
/usr/share/	Architecture-independent files.
/usr/src/	BSD and/or local source files.
/usr/X11R6/	X11R6 distribution executables, libraries, etc (optional).

/var/ Multi-purpose log, temporary, transient, and spool files. A memory-based file system is sometimes mounted at /var. This can be automated using the varmfs-related variables of rc.conf (or with an entry in /etc/fstab).

/var/log/ Miscellaneous system log files.

/var/mail/ User mailbox files.

/var/spool/ Miscellaneous printer and mail system spooling directories.

/var/tmp/ Contains temporary files. The files are usually preserved across a system reboot unless /var is a memory-based file system.

/var/yp/ NIS maps. **(11)**

System Configuration Files

/etc Generic system-specific configuration information.

/etc/defaults Default versions of system configuration files.

/etc/mail Extra sendmail configuration and other MTA configuration files.

/etc/ppp Configuration for both user- and kernel-ppp programs.

/etc/namedb Default location for named data. Normally named.conf and zone files are stored here.

/usr/local/etc Configuration files for installed applications. May contain per-application subdirectories.

/usr/local/etc/rc.d Scripts for installed applications.

/var/db Automatically generated system-specific database files, such as the package database and the locate database.

/etc/resolv.conf Configures how a FreeBSD system accesses the Internet Domain Name System (DNS).

/etc/hosts A simple text database which works in conjunction with DNS and NIS to provide host name to IP address mappings. **(12)**

Man Pages

A man page (short for manual page) is a form of online software documentation usually found on a Unix or Unix-like operating system. Topics covered include computer programs (including library and system calls), formal standards and conventions, and even abstract concepts. **(13)**

FreeBSD Man Pages can be accessed online at this address: http://www.freebsd.org/cgi/man.cgi

Download current FreeBSD Man Pages at this address: http://www.freebsd.org/cgi/man.cgi/faq.html

Basic FreeBSD Commands

Group	Command
Getting Help	man This command displays the manual page for a specified command.
	info This command an advanced man command that displays the improved manual pages.
	Apropos This command searches the manual page short descriptions for a specified keyword.
	whatis This command displays short man page descriptions.
	makewhatis This command creates the database for the whatis, appropos, and man commands.
File System Utilities	ls This command lists all the files in a specific directory.
	mkdir This command creates a directory within a filesystem structure.
	cd This command changes the current directory of the terminal shell.
	pwd This command displays the current directory that you are in using the command line terminal.
	chroot This command changes the root filesystem.
	cp This command copies a file.
	mv This command will move or rename files.
	rm This command deletes a file within a filesystem.
	rmdir This command deletes a directory within a filesystem.
	mkdir This command creates a directory within a filesystem.
	touch This command is used for changing the date on a file or for creating a blank file.

df

This command reports the amount of free disk space available on each partition.

link

This command creates a directory entry that associates a name with a file within a file system. This association ''links'' an existing directory entry to a new directory entry.

ln

This command creates links between files.

unlink

This command is used for deleting files. It is similiar to rm and rmdir.

chown

This command changes the owner and group of files.

chmod

This command changes the permissions of a specific files.

mount

This command informs the operating system when a file system is ready to use and sets the options relating to its access.

find

This command searches a given path for a file or folder.

whereis

This command searches the normal executable and man page locations for a specified file.

Finding Files

which

This command searches the locations in your PATH variable for a specified file.

locate

This command finds all filenames that match a specified query.

File Viewing

cat

This command displays the contents of a file to the screen.

more

This command paginates output and stops while waiting for data to fill the screen.

less

This command paginates output and has several features which "more" lacks.

od

This command allows the viewing binary files.

head

This command displays ten lines from the head (top) of a given file.

tail

This command displays last ten lines of the file.

File Editing

pico

pico is an easy to learn text editor originally designed for composing e-mail in Pine.

nano

nano is a clone of pico.

zile

zile is a lightweight and feature reduced clone of emacs.

vi

vi a powerful editor based on ex.

joe

joe is a full featured terminal-based screen editor.

emacs

emacs are a very powerful editor.

File Compression

gzip

This command compresses each individual file is compressed into a single file.

gunzip

This command uncompresses a file that was compressed with "gzip" or "compress".

zcat

This command will read files that are compressed with gzip without needing to uncompress them.

gzcat

This command will read files that are compressed without needing to uncompress them.

tar

This command archives data without compression.

pax

This command is similiar to the "tar" command but with different command-line syntax.

bzip2

This command is similar to "gzip"/"gunzip" but uses a different compression method.

bunzip2

This command decompresses all specified files. Compressed files that were not created by bzip2 will be detected and ignored.

zip

This command invokes an archive structure that compresses the members individually.

compress

This command reduces the size of files using adaptive coding.

File Analysing

file

This command displays the file type.

wc

This command tells you the number of lines, words and characters in a file.

cksum

This command displays the CRC checksum of files.

stat

This command displays the detailed status of a particular file or a file system.

Multiuser Commands

who

This command displays information about the users logged into a computer system.

finger

This command displays information about a user.

su

This command allows a system user to change the current user account using command line terminal access.

Self Information

whoami

This command displays you your current username.

groups

This command states the groups which the current user is a member of.

id

This command displays the same information as whoami and groups commands. This command also includes the user id (uid) and group id (gid) integers associated with the login.

tty

This command displays the terminal device that is assigned to your interactive login.

System Information

uptime

This command displays how long the computer has been running since its last reboot or power off.

uname

This command displays the system information such as hardware platform, system name, processor and operating system type.

dmesg

This command displays the messages from the kernel, since boot.

free

This command displays memory that is both used and free.

	vmstat This command displays a compact summary of overall system activity (processes, memory, and cpu information).
	top This command produces an ordered list of running processes selected by user-specified criteria, and updates it periodically.
	df This command reports the amount of free disk space available on each partition.
	hostname This command displays and sets the system hostname.
Networking	Ifconf This command is used for network interface configuration tool.
	ifdown This command takes a network interface down, placing it in a state where it cannot transmit or receive data.
	ifup This command brings a network interface up, making it available to transmit and receive data.
Process Management	nohup This command executes another command and makes it immune to any HUP (hangup) signals while the executed command is running.
	ps This command displays a list of current processes and their properties.
	kill This command is used to send termination signals to processes.
	pgrep This command is used to search and kill system processes.
	pidof This command will display Process ID (PID) of a task.
	killall This command will kill a process by name.
Devices	fuser This command displays what process is using a specific filesystem object such as a file or device.
	lsof This command lists all open files, and provides more detailed information than fuser.
	fstat This command lists all open files.

Miscellaneous	sync This command writes memory buffers to disk. echo This command displays the output status text to the screen or a file. cal This command displays the current months calendar. date This command displays the current date and time. time This command is used to determine the duration of execution of a particular command. from This command displays the names of those who sent you mail recently. mail This command allows you to read and write emails. clear This command clears the screen. PS1 This command is an environment variable that defines the shell prompt. **(14)**

File Systems

FreeBSD supports various file systems.

Native File Systems

UFS Unix File System.

UFS2 Unix File System 2 which is a continued development of the original Unix File System.

ZFS Zettabyte File System which is a next generation file system developed by Oracle Solaris. **(15)**

Supported File Systems

ext2 Second Extended File System was designed for use in the Linux kernel and provides improvements to the original ext file system.

ext3 Third Extended File System was designed for use in the Linux kernel and provides improvements to the ext2 file system.

ext4 Fourth Extended File System was designed for use in the Linux kernel and provides improvements to the ext3 file system.

Resier A general purpose file system developed by Hans Reiser for use in the Linux kernel. **(16)**

Packages And Ports

The FreeBSD Ports and Packages Collection offers a simple way for users and administrators to install applications. There are currently 24064 ports available.

The Ports Collection supports the latest release on the FreeBSD-CURRENT and FreeBSD-STABLE branches. Older releases are not supported and may or may not work correctly with an up-to-date ports collection. Over time, changes to the ports collection may rely on features that are not present in older releases. Wherever convenient, we try not to gratuitously break support for recent releases, but it is sometimes unavoidable. When this occurs, patches contributed by the user community to maintain support for older releases will usually be committed.

Ports

The FreeBSD Ports collection is a package management system for the FreeBSD operating system, providing an easy and consistent way of installing software packages.

Packages

Precompiled (binary) ports are called packages. A package can be obtained from the corresponding port with make package command; prebuilt packages are also available for download from the FreeBSD servers. A user can automatically install a package by passing the package name to the pkg install command. This downloads the appropriate package for the user's release version of FreeBSD, then installs the application along with any software dependencies it may have. By default, this command downloads packages from the main FreeBSD distribution site.
(17)

Package Benefits

* A compressed package tarball is typically smaller than the compressed tarball containing the source code for the application.

* Packages do not require compilation time. For large applications, such as Mozilla, KDE, or GNOME, this can be important on a slow system.

* Packages do not require any understanding of the process involved in compiling software on FreeBSD.

Port Benefits

* Packages are normally compiled with conservative options because they have to run on the maximum number of systems. By compiling from the port, one can change the compilation options.

* Some applications have compile-time options relating to which features are installed. For example, Apache can be configured with a wide variety of different built-in options.

* In some cases, multiple packages will exist for the same application to specify certain settings. For example, Ghostscript is available as a ghostscript package and a ghostscript-nox11 package, depending on whether or not Xorg is installed. Creating multiple packages rapidly becomes impossible if an application has more than one or two different compile-time options.

* The licensing conditions of some software forbid binary distribution. Such software must be distributed as source code that must be compiled by the end-user.

* Some people do not trust binary distributions or prefer to read through the source code in order to look for potential problems.

* Source code is needed in order to apply custom patches. **(18)**

Typically FreeBSD Ports tend to be more current than Packages.

Linux Emulation

FreeBSD provides 32-bit binary compatibility with Linux®, allowing users to install and run most 32-bit Linux® binaries on a FreeBSD system without having to first modify the binary. It has even been reported that, in some situations, 32-bit Linux® binaries perform better on FreeBSD than they do on Linux®.

Some Linux®-specific operating system features are not supported under FreeBSD. For example, Linux® binaries will not work on FreeBSD if they overly use i386™ specific calls, such as enabling virtual 8086 mode. In addition, 64-bit Linux® binaries are not supported at this time. **(19)**

linux_base-c6

This port contains packages from a near-minimal installation of CentOS 6 Linux. These packages, in conjunction with the Linux kernel module, form the basis of the Linux compatibility environment.

Note:

This port is only available for the i386/amd64 architecture (i386/32 bit mode).

If you want to run X11 applications, install the x11/linux-xorg-libs port.

http://www.freshports.org/emulators/linux_base-c6 **(20)**

The linux_base-c6 port provides a more current Linux compatibility base, more features and newer Linux kernel compared to the linux_base-f10.

Centos 6 was released on July of 2011 and Fedora 10 was released on November of 2008.

CentOS is a Linux distribution that attempts to provide a free, enterprise-class, community supported computing platform. The goals of CentOS are to be functionally compatible with its upstream source, Red Hat Enterprise Linux (RHEL). Red Hat Enterprise Linux (RHEL) is a Linux distribution developed by Red Hat and targeted toward the commercial market. **(21)** **(22)**

System Jails

The FreeBSD jail mechanism is an implementation of operating system-level virtualization that allows administrators to partition a FreeBSD-based computer system into several independent mini-systems called jails.

FreeBSD jails have two major goals:

Virtualization

Each jail is a virtual environment running on the host machine with its own files, processes, user and superuser accounts. From within a jailed process, the environment is (almost) indistinguishable from a real system.

Security

Each jail is sealed from the others, thus providing an additional level of security.

The FreeBSD jail mechanism restricts what processes in a jail can do in relation to the rest of the system. In effect, jailed processes are programs that are run separately. This can be used to execute untested code, or untrusted programs from unverified third parties, suppliers, untrusted users and untrusted websites. These processes are bound to specific IP addresses, and a jailed process cannot access or divert routing sockets. Internet sockets that allows direct sending and receiving of Internet Protocol packets without any protocol specific transport layer formatting are also disabled by default, but may be enabled by setting the security.jail.allow_raw_sockets sysctl option. Additionally, the interaction between processes that are not running in the same jail are restricted.

Virtualization

With a jail, it is possible to create various virtual machines, each of them having their own set of utilities installed and their own configuration. This makes it a safe way to try out software. For example, it is possible to run different versions or try different configurations of a web server package in different jails. Also, since the jail is limited to a narrow scope, the effects of a misconfiguration or mistake (even if done by the in-jail superuser) does not jeopardize the rest of the system's integrity. Since nothing has actually been modified outside of the jail, "changes" can be discarded by deleting the jail's copy of the directory tree.

Virtualization is valuable to service providers wishing to offer their users the ability to have custom configurations and yet keep the overall system easy to maintain. For example, two different customers could need different versions of the same software. Without jails, configuring multiple software versions in different directories and ensuring they do not encroach on each other isn't always possible or easy to maintain (e.g. XFree86 is notoriously hard to move around). Jails do permit software packages to view the system egoistically as if each package had the machine to itself. Jails can also have their own, independent, jailed superusers.

The FreeBSD jail does not however achieve true virtualization; it does not allow the virtual machines to run different kernel versions than that of the base system. All virtual servers share the same kernel and hence expose the same bugs and potential security holes. There is no support for clustering or process migration, so the host kernel and host computer is still a single point of failure for all virtual servers. It is possible to use jails to safely test new software, but not new kernels.

Security

FreeBSD jails are an effective way to increase the security of a server because of the separation between the jailed environment and the rest of the system (the other jails and the base system).

FreeBSD jails are limited in the following ways:

* Jailed processes cannot interact with processes in a different jail. For example, the ps command will only show the processes running in the jail.

* Modifying the running kernel by direct access and loading modules is prohibited. Modifying most sysctls and the securelevel is prohibited.

* Modifying the network configuration, including interfaces, interface or IP addresses, and the routing table, is prohibited. Accessing divert and routing sockets are also prohibited. Additionally raw sockets are disabled by default. A jail is bound only to specific IP addresses, and firewall rules cannot be changed.

* Mounting and unmounting filesystems is prohibited. Jails cannot access files above their root directory (i.e. a jail is chroot'ed).

* Jailed processes cannot create device nodes. **(23)**

System Security

Lynis

This is an auditing tool for Unix (specialists). It scans the system and available software, to detect security issues.

http://www.freshports.org/security/lynis/

Cops

This is a set of programs to check how secure your system is. It checks file and directory privileges, SUID programs, etc.

http://www.freshports.org/security/cops/

OpenBSM

This is an open source implementation of Sun's Basic Security Module (BSM) Audit API and file format.

http://www.freshports.org/security/openbsm-devel/

flawfinder

This examines source code looking for security weaknesses

http://www.freshports.org/security/flawfinder/

ClamTk

This is a GUI front-end for ClamAV.

http://www.freshports.org/security/clamtk/

Clam Antivirus

This is a command line virus scanner that has its database constantly updated. It also detects polymorphic viruses, scans compressed files and supported by AMaViS.

http://www.freshports.org/security/clamav/

References

FreshPorts has everything you want to know about FreeBSD software, ports, packages and applications.

http://www.freshports.org

The FreeBSD Handbook.

http://www.freebsd.org/doc/en/books/handbook

FreeBSD Security Information.

http://www.freebsd.org/security/

FreeBSD Security Advisories.

http://security.freebsd.org/advisories

FreeBSD announcement mail list.

This is the mailing list for people interested only in occasional announcements of significant FreeBSD events. This includes announcements about snapshots and other releases. It contains announcements of new FreeBSD capabilities.

http://lists.freebsd.org/mailman/listinfo/freebsd-announce

Citations

(1) http://en.wikipedia.org/wiki/Comparison_of_BSD_operating_systems

(2) http://en.wikipedia.org/wiki/FreeBSD

(3) http://simple.wikipedia.org/wiki/Linux

(4) http://en.wikipedia.org/wiki/Linux

(5) http://www.freebsd.org/releases/

(6) http://www.freebsd.org/doc/en/books/handbook/current-stable.html

(7) http://en.wikibooks.org/wiki/Guide_to_Unix/Why_Unix-like

(8) http://en.wikibooks.org/wiki/A_Quick_Introduction_to_Unix/Components

(9) http://en.wikibooks.org/wiki/A_Quick_Introduction_to_Unix/Files_and_Processes

(10) http://en.wikipedia.org/wiki/FreeBSD

(11) http://www.freebsd.org/doc/handbook/dirstructure.html

(12) http://www.freebsd.org/doc/en/books/handbook/configtuning-configfiles.html

(13) http://en.wikipedia.org/wiki/Man_page

(14) http://en.wikibooks.org/wiki/Guide_to_Unix/Commands

(15) http://www.freebsd.org/doc/handbook/filesystems.html

(16) http://www.freebsd.org/doc/handbook/filesystems-linux.html

(17) http://en.wikipedia.org/wiki/FreeBSD_Ports

(18) http://www.freebsd.org/doc/handbook/ports-overview.html

(19) http://www.freebsd.org/doc/handbook/linuxemu.html

(20) https://svnweb.freebsd.org/ports/head/emulators/linux_base-c6/pkg-descr?revision=HEAD

(21) http://en.wikipedia.org/wiki/CentOS

(22) http://en.wikipedia.org/wiki/Red_Hat_Enterprise_Linux

(23) http://en.wikipedia.org/wiki/FreeBSD_jail